Math 131 notes

Jason Riedy

15 September, 2008

Contents

Notes also available as PDF.

1 Review

Structure of the upcoming test:

- Ten questions. Chose ***six*** and solve them.
- Thus expect about seven minutes per question.
- Remember to read and answer the ***entire*** question.
- Closed book, *etc.* Calculators are fine but not necessary.

- Bring scratch paper and paper for writing up your results. Separately.
- Answers and explanations need to be indicated clearly.
- No questions are intended to be "trick" questions.
- Will cover the following topics:
  - inductive v. deductive reasoning with sequences,
  - problem solving,
  - set theory,
  - symbolic logic.
- Remember that solutions for the homework problems are available on-line: `http://jriedy.users.sonic.net/VI/math131-f08/`.

2 Inductive and deductive reasoning

Two primary forms of reasoning:

inductive Working from examples and intuiting how to extend them. Inductive reasoning does not prove anything.

deductive Extending hypotheses with rules to reach a conclusion. Deductive reasoning generates proofs (even if simple).

An example of *inductive reasoning*:

> It has been sunny all week, so it will be sunny tomorrow.

There are no explicit rules or assumptions. We just assume the pattern continues.

An example of *deductive reasoning*:

> The weather forecasts state that if the storm turns northward, it will rain tomorrow. The storm has turned northward, so it will rain tomorrow.

This sets up a rule from the weather forecast. Then the rule is applied to data, the storm turning northward, to reach a conclusion about tomorrow.

There rarely are clean-cut distinctions. Consider extending the sequence

$$35, 45, 55, \ldots$$

Reasoning *inductively*, we might assume that the numbers jump by ten. The next number will be 65, then 75, and so on.

Reasoning *deductively*, we assume this is an arithmetic sequence starting at 35 with an increment of 10. Under this assumption, the next two numbers will be $35 + 10 \cdot 3 = 65$ and $35 + 10 \cdot 4 = 75$.

The difference between the two forms is subtle. Deductive reasoning sets up explicit rules. The rules themselves may be discovered inductively, but making the rules specific and applying them carefully renders the result deductive.

3 Problem solving

Pólya's principles:

1. Understand the problem.
2. Devise a plan.
3. Carry out the plan.
4. Look back at your solution.

This is not a simple 1-2-3-4 recipe. Understanding the problem may include playing with little plans, or trying to carry out a plan may lead you back to trying to understand the problem.

3.1 Understand the problem

- Read the **entire** problem.

  Read the **whole** problem.

  Read **all of the** problem.

  One comment about the homeworks: Most people answer only part of any given problem.

- Determine what you **have** and what you **want**.

  To indicate an answer clearly to someone else (like me), you need to know what the answer is.

- Consider rephrasing the problem, either in English or symbolically.

  Rephrasing the problem may help you remember solution methods.

- Try some examples.

  This is close to devising a plan. Sometimes you may stumble upon an answer.

- Look for relationships between the data.

  Examples may help find relationships. The relationships may help you decide on a plan. Mathematics is about relationships between different entities; symbolic mathematics helps abstract away the entities themselves.

3.2 Devise a plan

Sometimes plans are "trivial," or so simple it seems pointless to make them specific. But write it out anyways. Often the act of putting a plan into words helps find flaws in the plan.

Try to devise a plan that you can check along the way. The earlier you detect a problem, the easier you can deal with it.

Some plans we've considered:

- Guessing and checking.

  Try a few combinations of the data. See what falls out. This is good for finding relationships and understanding the problem.

- Searching using a list.

  If you know the answer lies in some range, you can search that range systematically by building a list.

- Finding patterns.

  When trying examples, keep an eye open for patterns. Sometimes the patterns lead directly to a solution, and sometimes they help to break a problem into smaller pieces.

- Following dependencies / working backwards.

  Be sure to understand what results depend on which data. Look for dependencies in the problem. Sometimes pushing the data you have through all the dependencies will break the problem into simpler subproblems.

3.3 Carry out the plan

Attention to detail is critial here.

When building a list, be sure to carry out a well-defined procedure. Or when looking for patterns, be systematic in the examples you try. Don't jump around randomly.

3.4 Look back at your solution

Can you check your result? Sometimes trying to check reveals new relationships that could lead to a better solution.

Think about how your solution could help with other problems.

4 Sequences

A sequence is an ordered list of numbers. Two common kinds are:

arithmetic Adds a constant increment at each step.

geometric Multiplies by a constant at each step.

One method for extending a sequence is through **successive differences**. Consider the sequence

$$11 \quad 22 \quad 39 \quad 64 \quad \ldots$$

To compute the next term, form differences until you find a constant column:

i	A_i	$\Delta_i^{(1)} = A_i - A_{i-1}$	$\Delta_i^{(2)} = \Delta_i^{(1)} - \Delta_{i-1}^{(1)}$
1	11		
2	22	11	
3	39	17	6
4	62	23	6
5	**91**	**29**	**6**

5 Set theory

set An **unordered** collection of **unique** elements.

You can write a set by listing its entries, $\{1, 2, 3, 4\}$, or through set builder notation, $\{x \mid x \text{ is a positive integer}, x < 5\}$.

empty set The **unique** set with no elements: $\{\}$ or $\emptyset$. Be sure to know the relations between the empty set and other sets, and also how the empty set behaves in operations.

element of You write $x \in A$ to state that x is an element of A. The symbol is not an "E" but is almost a Greek ϵ. Think of a pitchfork.

Note that $1 \in \{1, 2\}$ and $\{1\} \in \{\{1\}, \{2\}\}$, but $\{1\} \notin \{1, 2\}$.

subset Given two sets A and B, $A \subset B$ if every element of A is also an element of B. So $A \subset B$ is equivalent to $x \in A \to x \in B$.

One implication is that $\emptyset \subset A$ for all sets A. This statement is **vacuously** true.

Here $\{1\} \subset \{1, 2\}$ and $\{1\} \not\subset \{\{1\}, \{2\}\}$.

superset Given two sets A and B, $B \supset A$ if every element of A is also an element of B.

proper subset or superset A subset or superset relation is **proper** if it implies the sets are not equal. An equivalent symbolic logic statement would be $(x \in A \rightarrow x \in B) \wedge (\exists x \in B : x \notin A)$.

Venn diagram A blobby diagram useful for illustrating operations and relations between two or three sets.

union $A \cup B = \{x \mid x \in A \vee x \in B\}$. The union contains all elements of both sets.

intersection $A \cap B = \{x \mid x \in A \wedge x \in B\}$. The intersection contains only those elements that exist in both sets.

set difference $A \setminus B = \{x \mid x \in A \wedge x \notin B\}$. The set difference contains elements of the first set that are **not** in the second set. It cannot contain any elements of the second set.

You can use symbolic logic to write the result of multiple operations.

$$\begin{aligned}(A \cap B) \cup C &= \{x \mid x \in A \wedge x \in B\} \cup C \\ &= \{x \mid (x \in A \wedge x \in B) \vee x \in C\}.\end{aligned}$$

6 Symbolic logic

logical statement Some clear statement that is either true or false.

Some different ways of writing true or false are acceptable:

true	false
T	F
1	**0**
$\top$	$\bot$

The test's questions use 1 and 0.

logical variable A variable standing for some logical statement. Common variables are p, q, r.

truth table A systematic listing of all possible input truth values for an expression.

negation True when the variable is false and false when the variable is true. Will be written $\neg p$.

and True only when all variables are true. Will be written $p \wedge q$.

or False only when all variables are false. Will be written $p \vee q$.

equivalence The logical form of equality. Will be written $p \equiv q$.

conditional If p then q. True whenever true implies true or when false implies anything. Will be written $p \rightarrow q$.

tautology A statement that always is true. Will be written $\models$, as in $\models p \vee \neg p$. This is just for emphasis; there is no real difference with $(p \vee \neg p) \equiv 1$.

A truth table defining four operations above:

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \rightarrow q$
1	1	0	1	1	1
1	0	0	0	1	0
0	1	1	0	1	1
0	0	1	0	0	1

Note that $\neg p$ did not need all four lines. It does not depend on q and has the same value regardless of whether q is false or true.

6.1 From truth tables to functions

Consider the truth table:

p	q	$f(p,q)$
1	1	1
1	0	0
0	1	1
0	0	0

We can derive an expression for $f(p,q)$ in two ways. Obviously, any expressions must simplify to q.

One method is to work from the true values. You *or* together *and* expressions. There is one *and* expression per true value. In this case, we have $\models f(p,q) \equiv (p \wedge q) \vee (\neg p \wedge q)$. Pulling out the q, this simplifies $\models f(p,q) \equiv (p \vee \neg q) \wedge q \equiv 1 \wedge q \equiv q$.

The other method is to work from the false values. You *and* together *or* expressions. There is one *or* expression per false value. Here, $\models f(p,q) \equiv (\neg p \vee q) \wedge (p \vee q)$. Pulling out q again, $\models f(p,q) \equiv (\neg p \wedge p) \vee q \equiv 0 \vee q \equiv q$.

6.2 De Morgan's laws and forms of conditionals

De Morgan's laws are two very useful methods for negating terms symbolically:

$$\models \neg(p \vee q) \equiv \neg p \wedge \neg q, \text{ and}$$
$$\models \neg(p \wedge q) \equiv \neg p \vee \neg q.$$

As an example, consider negating the conditional. Use the equivalent form $\models p \to q \equiv \neg p \vee q$. Then

$$\begin{aligned} \neg(p \to q) &= \neg(\neg p \vee q) \\ &= \neg(\neg p) \wedge \neg q \\ &= p \wedge \neg q. \end{aligned}$$

So the negation of a conditional is **not** a conditional itself.

There are four related forms of conditional:

conditional $p \to q$: If you grew up in Alaska, you have seen snow.

inverse $\neg p \to \neg q$: If you did not grow up in Alaska, you have not seen snow.

converse $q \to p$: If you have seen snow, you grew up in Alaska.

contrapositive $\neg q \to \neg p$: If you have not seen snow, you did not grow up in Alaska.

Only the **contrapositive** has the same meaning as the original conditional, so $\models p \to q \equiv \neg q \to \neg p$.

The **converse** and **inverse** are related to each other but are **not** equivalent to the original conditional. The inverse is the contrapositive of the converse: $\models q \to p \equiv \neg p \to \neg q$

6.3 Quantifiers

quantifier A statement regarding some or all possible entries of some set.

- **existential** Declares that some entry exists, so $\exists x : x \in A$ states that A is not empty.
- **universal** Declares some property is true for every value. So $\forall x \in A : x \in B$ is another way of writing $A \subset B$.

predicate Or **property**. A symbolic way of expressing that some property holds. For example, understands(s, t) may state that student s understands topic t. A less obtuse but still acceptable statement for a simple predicate is just "s understands q."

The translation of phrases from English to quantified symbolic logic can be tricky.

> Almost every student understands all symbolic logic topics.

can translate to

$$\exists s\,\forall t : \neg\,\text{understands}(s,t)$$

because we don't measure *how many* but rather that there is or is not one.

6.4 Nesting and negating quantifiers

Nested quantifiers are not operators. Each quantifier applies to the entire remaining statement. $\forall s \exists t$ states that for every s, there exists a t **for that s**. Meanwhile, $\exists t \forall s$ states that there exists one single t for every and all s.

Two rules for negating quantifiers:

> $\neg\forall s : P(s)$ is the same as $\exists s : \neg P(s)$, and
>
> $\neg\exists s : P(s)$ is the same as $\forall s : \neg P(s)$.

So saying "not all" is the same as "there exists one for not", and saying "there does not exist" is the same as "for all, not".

As an example, we negate the statement above,

> Almost every student understands all symbolic logic topics.

and its translation

$$\exists s\,\forall t : \neg\,\text{understands}(s,t).$$

The symbolic negation is

$$\begin{aligned}\neg(\exists s\,\forall t : \neg\,\text{understands}(s,t)) &= \forall s\,\neg(\forall t : \neg\,\text{understands}(s,t)) \\ &= \forall s\,\exists t : \neg(\neg\,\text{understands}(s,t)) \\ &= \forall s\,\exists t : \text{understands}(s,t).\end{aligned}$$

Translating back to English,

> All students understand some symbolic logic topic.

It may be true that no two students understand the *same* topic, but every student understands some topic.